CODEFORCES
Sponsored by TON

CSE221 Assignment 03 Spring 2025

A. Count the Inversion

time limit per test: 1 second
memory limit per test: 256 megabytes

Here is a Pseudocode of the Merge Sort Algorithm.

```
def merge(a, b):
    # write your code here
    # a and b are two sorted list
    # merge function will return a sorted list after merging a and b

def mergeSort(arr):
    if len(arr) <= 1:
        return arr
    else:
        mid = len(arr)//2
        a1 = mergeSort(...........)  # write the parameter
        a2 = mergeSort(...........)  # write the parameter
        return merge(a1, a2)         # complete the merge function above
```

Now, you are given an array ***A*** of size ***N*** of ***N*** distinct integers. It is guaranteed that the array A contains a permutation of integers from 1 to N (i.e., every integer from 1 to N appears exactly once).

1. Count the number of inversions in the given array.
2. Sort the array in non-decreasing order.

An inversion is a pair (i, j) where $i < j$ and $A[i] > A[j]$.

Input

The first line contains an integer ***N*** $(1 \leq N \leq 10^5)$ — denoting the length of the list.

In the next line, there will be N integers $a_1, a_2, a_3 \ldots a_n$ $(1 \leq a_i \leq N)$ separated by spaces.

Output

In the first line, print the total number of inversions in the given array. In the next line, print the array in non-decreasing order.

Examples

input

```
5
1 2 5 4 3
```

output

```
3
1 2 3 4 5
```

input

```
5
1 2 3 4 5
```

output

```
0
1 2 3 4 5
```

input

```
5
5 4 3 2 1
```

output

```
10
1 2 3 4 5
```

input

```
7
6 4 2 5 7 3 1
```

output

```
14
1 2 3 4 5 6 7
```

Note

In the first example, the inversions are pair $(3, 4), (3, 5)$ and $(1, 5)$

In the second example, there are no inversions.

In the third example, every pair of i, j where $i < j$, we have $A[i] > A[j]$. Hence, All 10 such pairs are inversions.

B. Pair Maximization

time limit per test: 1 second
memory limit per test: 256 megabytes

you are given an array ***A*** of size ***N***. You have to choose two indices i and j such that $1 \leq i < j \leq N$ and $A[i] + A[j]^2$ is the maximum possible. Here, we are considering 1-based indexing. Come up with a divide and conquer approach to solve the problem.

Input

The first line contains an integer ***N*** $(2 \leq N \leq 10^5)$ — denoting the length of the list.

In the next line, there will be N integers $A_1, A_2, A_3 \ldots A_n$ $(-10^9 \leq A_i \leq 10^9)$ separated by spaces.

Output

Print a single integer - which denotes the maximum possible value of $A[i] + A[j]^2$.

Examples

input

```
5
4 3 1 5 6
```

output

```
41
```

input

```
5
4 3 1 -9 6
```

output

```
85
```

C. Fast MOD Drift

time limit per test: 1 second
memory limit per test: 256 megabytes

You are given two integers ***a*** and ***b***. Calculate $a^b \ mod \ 107$.

Input

The input file contains two integers ***a*** $(1 \leq a \leq 10^4)$ and ***b*** $(1 \leq b \leq 10^{12})$.

Output

Print one integer — the result of $a^b \ mod \ 107$.

Examples

input

```
100 3
```

output

```
85
```

input

```
100 5
```

output

```
99
```

input

```
10000 1000000000000
```

output

```
27
```

D. Fast MOD Drift Revisited

time limit per test: 2.5 seconds

memory limit per test: 256 megabytes

You are given three integers ***a***, ***n*** and ***m***. Calculate $(a^1 + a^2 + \ldots + a^n) \,\%\, m$.

Input

The first line contains an integer ***T*** $(1 \leq T \leq 10^5)$ — total numbers of test cases.

In each of the next T test cases, there are three integers ***a*** $(1 \leq a \leq 10^6)$, ***n*** $(1 \leq n \leq 10^{12})$ and $(1 \leq m \leq 10^9)$

Output

Print one integer — the result of $(a^1 + a^2 + \ldots + a^n) \,\%\, m$.

Example

input

```
3
2 5 1000
2 9 1000
1 100 30
```

output

```
62
22
10
```

E. Ordering Binary Tree

time limit per test: 1 second

memory limit per test: 256 megabytes

you are given an array ***A*** of size ***N*** in **increasing** order. Find an order of these N integers such that, if these integers are inserted into a Binary Search Tree (BST) one by one, the height of the resulting BST is minimized.

A Binary Search Tree is a binary tree in which each node has at most two children, referred to as the left and right child. For any node, all elements in the left subtree are smaller than the node's value, and all elements in the right subtree are greater than the node's value.

The height of a Binary Search Tree is defined as the maximum depth among all the nodes in the tree.

Note: All the elements in the array A are guaranteed to be unique. In other words, $A_i \neq A_j$ if $i \neq j$.

Input

The first line contains an integer ***N*** $(1 \leq N \leq 10^5)$ — denoting the length of the list.

In the next line, there will be N integers $a_1, a_2, a_3 \ldots a_n$ $(1 \leq a_i \leq 10^9)$ in non-descending order separated by spaces.

Output

Output the order of the elements such that when inserted into a Binary Search Tree, the height of the tree is minimized. If there are multiple such orders then find any of them.

Example

input

```
5
1 2 3 4 5
```

output

```
3 1 2 4 5
```

F. 220 Trees

time limit per test: 1 second

memory limit per test: 256 megabytes

There is a Binary Tree with ***N*** nodes. You are given the in–order and pre-order traversals of the tree. Your task is to determine the post-order traversal of the tree.

Input

The first line contains an integer N $(1 \leq N \leq 1000)$ — the number of nodes in the binary tree.

In the next line, there will be N integers $a_1, a_2, a_3 \ldots a_n$ $(1 \leq a_i \leq N)$ separated by spaces – representing the in-order traversal of the tree.

The following line, there will be N integers $b_1, b_2, b_3 \ldots b_n$ $(1 \leq b_i \leq N)$ separated by spaces – representing the pre-order traversal of the tree.

Output

Print N space-separated integers representing the post-order traversal of the binary tree.

Example

input

```
5
4 2 5 1 3
1 2 4 5 3
```

output

```
4 5 2 3 1
```